NRL Report 7351

Software Simulation of an Associative Processor

JOHN E. SHORE

Information Systems Group
Office of the Associate Director of Research
for Electronics

December 16, 1971

NAVAL RESEARCH LABORATORY
Washington, D.C.

CONTENTS

ACRONYMS

AADC	Advanced Avionics Digital Computer
A&C	Arithmetic and control
AMAR	Associative memory address register
AMOB	Associative memory output buffer
AP	Associative processor
APC	Associative processor control
APL	AP length
APM	Associative processor memory
APMW	APM width
CM	Clock mask
CMD	Controller memory direct
CMEM	Controller memory
CMI	Controller memory indirect
CML	Controller memory length
CMSK	Controller mask register
CMW	Controller memory width
COMP	Comparand
CSDR	Controller syllable definition register
HCR	Hoizontal control registers
IB	Input buffer
IM	Instruction memory
NRSP	Number of ROB's set previously register
OB	Output buffer
RB	Response bit
ROB	Response output bit
RRN	Response resolution network
RS	Response store
RSW	RS width
SC1	Search control one
SC2	Search control two
SDR	Syllable definition register
SREG	Special registers

Software Simulation of an Associative Processor

JOHN E. SHORE

Information Systems Group
Office of the Associate Director of Research for Electronics

Abstract: A software simulation of an associative processor (AP) has been developed in Fortran as a tool for use in some research projects at NRL. The particular AP design being simulated is that described in NRL Report 7348, a design oriented towards the requirements of the Advanced Avionic Digital Computer (AADC) under development by the Naval 'Air Systems Command.

The simulated AP is driven by a simulated associative processor controller (APC). The APC is a simple, sequential computer with an instruction memory, a data memory, an arithmetic and control section, and a set of special, AP associated registers. The entire AP/APC simulation is at the bit level with the exception of the APC instruction memory, which contains a mnemonic APC code written by the user.

In this report the overall simulation is described and instructions for its use are given. A complete example is included.

1. Introduction

A software simulation of an associative processor (AP) has been developed as a tool for use in a number of research projects underway at NRL. The particular associative processor being simulated has evolved from an earlier design based on requirements for ELINT (electronic intelligence or electromagnetic intercept) signal processing (1). The additional flexibility and capability of the present design are the result of orienting the AP toward the requirements of the Advanced Avionics Digital Computer (AADC) under development by the Naval Air Systems Command. The reader is assumed to be familiar with the present design of the AP, which is described in NRL Report 7348 (2).

As a research tool the simulation program was written with three applications in mind: the verification of bit-level AP transfer functions, the investigation of AP control structures, and the development of realistic AP algorithms (including all necessary bookkeeping). The program is written in Fortran for the CDC 3800 computer. The main program and its 24 subroutines take up 113,300 words of core, not counting system and library routines.

The next section of this report gives an overview of the simulation. The instruction set for the AP and its controller is described in Section 3. The program deck organization necessary

Note: The work of this report was performed when the author was with the Intercept and Signal Processing Branch, Communication Science Division.

NRL Problem R06-41: Projects NavAirSysCom WF08-151-702 and WF15-241-601/602. This is an interim report on a continuing NRL Problem. Manuscript submitted September 20, 1971.

to run the simulation is described in Section 4. Program output is discussed in Section 5. A complete example is given in Section 6. The printout from this sample program is included as Appendix A.

2. Simulation Overview

The simulation is divided into two parts: the associative processor (AP) and the associative processor controller (APC). A block diagram showing the relationship between AP and APC components is given in Fig. 1. The characteristics of the APC were chosen to provide a means of exercising the AP under stored program control. The details of the APC simulation should not be considered as a design for an actual APC.

The simulated APC consists of four sections:

- Instruction memory (IM) – contains the APC program,
- Controller memory (CMEM) – contains data,
- Arithmetic and control section (A&C) – executes programs stored in IM and sets AP control lines,
- Special registers (SREG) – registers with special functions in the APC and AP control structures.

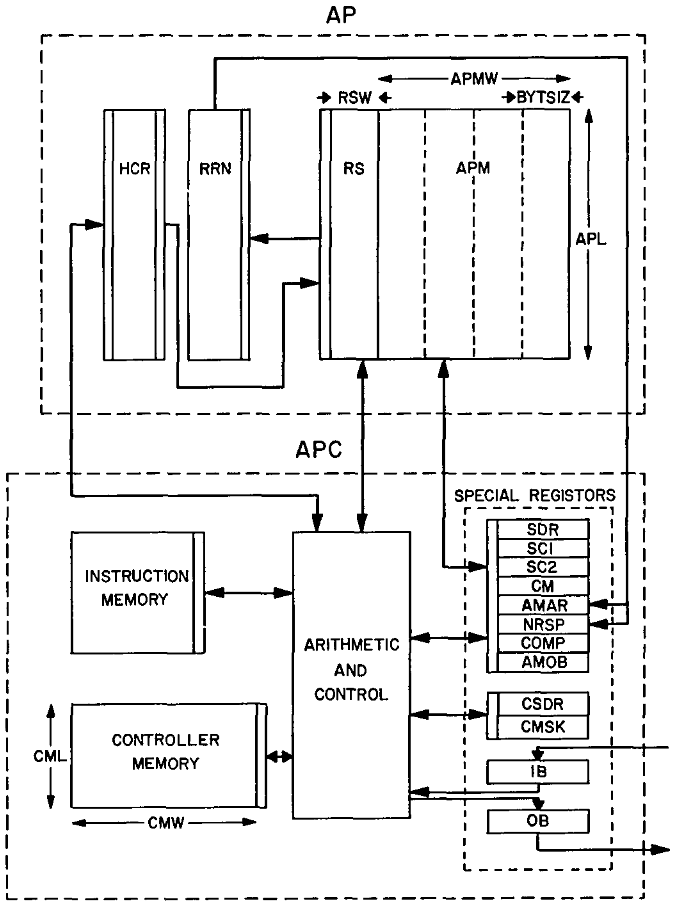


Fig. 1 – Simulation overview

The latter three are simulated at the bit level, and the contents of the IM are mnemonic dodes which may be considered as microprograms.

For programming convenience APC instructions access CMEM and SREG directly without the use of address registers and separate read/write instructions. CMEM and SREG should be considered as the same type of memory. The special registers have been separated in the simulation and named according to their special purpose only for APC programming convenience. The use of each of these registers is briefly as follows:

Controller syllable definition register (CSDR) – The CSDR divides the APC into syllables. A "1" in a CSDR bit position means that the bit position is the first bit (counting right to left) of an APC syllable. In an add or subtract operation, carry propagations are inhibited across syllable boundaries defined by the CSDR.

Controller mask register (CMSK) – The CMSK has two masking functions within the APC. In the add and subtract operations the second operand (OPR2 in the discussion of Sections 3.2.4) is masked (logically ANDed) with the CMSK. Also, when information is gated to any CMEM or SREG location (except the CMSK itself), the replacement is made only for those bits in the destination location which have a corresponding "1" in the CMSK.

Input buffer (IB) – The IB provides the means for obtaining external data. When any instruction gates a value from the IB, that is, when the IB is specified as an operand, its contents are replaced by those of a card which is read in by the host computer system. The format of each such card is a string of octal integers right-justified in columns 1 through 43. APC instructions may not gate information onto the IB.

Output buffer (OB) – The OB provides the means of outputting data under APC control. Whenever an instruction gates information onto the OB, the new OB contents are printed on the output of the host computer system one APC cycle later.

Syllable definition register (SDR) – The SDR is the AP analog of the CSDR. It divides the AP into syllables. There is one bit in the SDR for each AP byte, and a "1" in a SDR bit position means that the corresponding AP byte is the first byte of an AP syllable.

Search control one and search control two – The SC1 and SC2 specify the byte-variable AP seach criteria. The code for each SC1/SC2 bit, corresponding to one AP byte, is reviewed in Table 1. The SC1/SC2 search specification must not change within each syllable defined by the SDR. The SC1 extends over the response store (RS) section of the AP and has one bit for each RS bit. The search code for the RS section is reviewed in Table 2. The SC1 is also used in conjunction with the comparand register (COMP) to define the response bit (RB) for an AP search instruction. The code for this is also reviewed in Table 2.

Table 1
Search Specification Codes for One AP Byte Slice

Input		Type of Search
SC1	SC2	
0	1	Greater than or equal to X inputs
1	0	Less than or equal to X inputs
1	1	Exact match to X inputs
0	0	Anything will do (don't care)

Table 2
Search Specification Code and RB
Specification Code for One RS Bit Slice

Input		Type of Search
COMP	SC1	
–	1	Exact match to COMP
0	0	Don't care
1	0	Don't care; identifies this bit slice as the RB

Clock mask (CM) – The CM is used to inhibit the clock from entire AP byte slices. A "1" in a CM bit inhibits the corresponding AP byte from being clocked. The clock is applied to AP words on a selective basis by the RS according to the RS control structure. The CM provides the capability to prevent individual bytes within these selected words from being clocked.

Associative memory address register (AMAR) – The AMAR is the address register for conventional AP operations. Also, upon command, the response resolution network (RRN) gates onto the AMAR the largest address of any word whose response output bit (ROB) is set. This provides a means of processing search responses.

Number of ROB's set previously register (NRSP) – At each APC clock the NRSP is loaded with the total number of ROB's that were set before the clock. This provides additional capability in the processing of search responses. The operation of the NRSP is probably unrealistic from a hardware viewpoint; a more realistic NRSP might contain an approximation to the number of responses, perhaps obtained by passing the ROB outputs through a summation amplifier and making an analog-to-digital conversion on the amplifier output. Since the principal APC programming use of NRSP is to provide an indication of *no* responses, an indication that the more realistic NRSP would also give reliably, the ideal NRSP has been incorporated. This has the additional advantage of permitting the APC simulation to output precise information on search responses. APC instructions do not gate values onto the NRSP.

Comparand (COMP) – The COMP is the data input register of the AP. It has one bit for each AP bit slice (including the RS). The COMP outputs are connected to the X inputs of the first AP word.

Associative memory output buffer (AMOB) – The AMOB is the data output register of the AP. Depending on the AP instruction, it may be loaded with the result of a conventional AP read or with the data present on the $\hat{X}$ outputs of the bottom AP word.

Each APC instruction contains a section specifying AP control inputs which are passed directly to the AP by the A&C. The AP simulation has four sections (Fig. 1): associative processor memory (APM), response store (RS), response resolution network (RRN), and horizontal control registers (HCR).

The APM and RS are simulated at the bit level by sequencing through single-bit transfer functions. The RRN is simulated functionally. The APM, RS, and RRN are described in NRL Report 7348 (2).

The HCR is a set of five bit registers that are used to provide the A_i and B_i inputs to each AP word when the normal RS control structure is overridden by the APC. Although the HCR is not likely to be part of an actual APC, it is included in the simulation to aid in the investigation of control structures and algorithm development. The HCR is loaded directly by the APC and is simulated at the bit level.

Various CMEM and AP size parameters are simulation variables that are user inputs (Fig. 1 and Table 3). Lengths of special registers are automatically adjusted according to these inputs.

The mnemonic instruction language (contents of the IM) contains a number of non-APC-executable instructions for printing out APC and AP contents as well as various execution statistics that are accumulated by the simulation program. The simulation does not keep track of APC program execution time (all APC/AP instructions are assumed to execute in equal time).

Table 3
APC/AP Simulation Variables

Variable	Description	Limitation
APL	Number of AP words	APL $\leq$ 512
APMW	Total number of bits in the non-RS section of each AP word	–
RSW	Number of bits in the RS section	RSW + APMW $\leq$ 128
BYTSIZ	Number of bits in one AP byte	$\frac{\text{APMW}}{\text{BYTSIZ}}$ = integer
CMW	Number of bits in each CM word	CMW $\leq$ 128
CML	Number of controller memory words	CML $\leq$ 1024

3. Simulation Instruction Set

Each APC/AP instruction is placed on one computer card. Columns 1 through 4 contain the instruction label. The APC portion of the instruction is contained in columns 5 through 44, and the AP portion is contained in columns 45 through 71. If the first four columns contain only a left-justified asterisk ("* ") the contents of the card are printed as a comment in the APC/AP program listing and otherwise ignored by the simulation.

3.1 Instruction Label

The first four columns of an instruction card are reserved for an alphanumeric label used to identify the instruction as the destination of a control transfer. These four columns may be left blank if no label is required.

3.2 APC Instructions

The APC instruction is contained in columns 5 through 44. It is organized in ten fields of four columns each. The contents of each field must be right justified. The APC instruction must begin in the first field (columns 5 through 8) and fill as many successive fields as required, up to nine of the ten fields. There should be no blank fields within the instruction. If the instruction card contains an AP portion in columns 45 through 71, the field following the final field used in specifying the APC instruction should be left blank. If the instruction has no AP portion, the field following the final nonblank APC field should contain a right-justified comma ("   ,"). If the card contains only an AP portion, the entire APC portion should be left blank. A complete review of APC instructions is given in Table 4.

3.2.1 Memory References

The majority of APC instructions involve one or more references to the contents of locations in CMEM, SREG, or HCR.

3.2.1.1 References to Controller Memory

A reference to a location in CMEM requires two APC instruction fields and may be either direct (CMD) or indirect (CMI). A direct reference is made as

" CMD   n",

which specifies location n in CMEM ($n \leq$ CML). An indirect reference is made as

" CMI   n",

which specifies the contents of the CMEM word whose address is to be found in CMEM location n. The first location in memory is given by $n = 0$.

3.2.1.2 References to Special Registers

A reference to a location in SREG requires either one or two APC instruction fields. A direct or indirect reference to SREG may be made as

" SRD   n"

or

" SRI   n"

respectively, where the n values are listed at the right in Table 4 and correspond to the order shown in Fig. 1. These references are similar to those specifying locations in CMEM. A direct reference to a location in SREG may also be made by use of an alphanumeric acronym which requires only one APC instruction field. Thus, for example, the references " SRD   6" and "COMP" are equivalent. The properties of each special register were discussed in Section 2.

3.2.1.3 References to Horizontal Control Registers

A reference to a location in HCR requires two APC instruction fields. The reference may be direct only and is given by

" HCR   n".

Table 4
Summary of the APC and Utility Instruction Set

Class	Mnemonic	No. of APC Instruction Fields	Meaning
APC Instructions			
Memory references	CMD n	2	Controller memory direct
	CMI n	2	Controller memory indirect
	HCR n	2	Horizontal control register
	SRD n	2	Special register direct
	SRI n	2	Special register indirect
	SDR	1	Syllable definition register (SREG location 0)
	SC1	1	Search control one (SREG location 1)
	SC2	1	Search control two (SREG location 2)
	CM	1	Clock mask (SREG location 3)
	AMAR	1	Associative memory address register (SREG location 4)
	NRSP	1	Number of responses (SREG location 5)
	COMP	1	Comparand (SREG location 6)
	AMOB	1	Associative memory output buffer (SREG location 7)
	CSDR	1	Controller syllable definition register (SREG location 8)
	CMSK	1	Controller mask (SREG location 9)
	IB	1	Input buffer (SREG location 10)
	OB	1	Output buffer (SREG location 11)
Immediate constant	IMD	1	Immediate constant
Unconditional transfer	GOTO	1	GO TO (followed by label of destination instruction)
Termination	STOP	1	Terminate execution
Short operations	=	1	Store (location of store follows to the right)
	LS n	2	Left shift and store
	RS n	2	Right shift and store
	JMPZ	1	Jump if zero (followed by label of destination instruction)
Long operations	AND	1	AND
	OR	1	OR
	NOT	1	NOT
	EOR	1	EXCLUSIVE OR
	+	1	Add
	−	1	Subtract
Utility Instructions			
	DMCM	1	Dump CMEM
	DMSR	1	Dump SREG
	DMAP	1	Dump AP
	DMHR	1	Dump HCR
	DMAL	1	Dump all

	STAT m	2	Output statistics list m: m = 1 all statistics
			m = 2 basic program statistics
			m = 3 detailed APC statistics
			m = 4 detailed AP statistics

3.2.2 Immediate Constants

APC instructions may also reference as an operand a constant provided with the instruction. This is called an immediate constant and is referenced by

" IMD".

If an APC instruction contains an " IMD" reference, the constant should appear on a separate card immediately following the instruction. The format is a string of octal integers right justified in columns 1 through 43.

3.2.3 Short Instructions

Short APC instructions involve one operand, one operation, and one destination. An operand may be an immediate constant or any location in CMEM, SREG, or HCR.

Four operations may be used in a short instruction: direct store, left shift and store, right shift and store, and jump if zero. The definition and format of these short operations is given in Table 5.

If the operation involves a control transfer ("JMPZ"), the destination is a location in instruction memory (IM). In this case the destination requires one APC instruction field which is filled with the label of the instruction to which control is to be transferred (Section 3.1). For the other short operations, all involving a store, the destination is a location in CMEM, SREG, or HCR and is referenced as described in Section 3.2.1.

The format of a short instruction is an operand (OPR) followed by a short operation (SOP) followed by a destination (DEST), symbolically OPR.SOP.DEST. Thus

```
column
  1
  ↓
"A    CMD   18   =COMP"
```

Table 5
Short APC Instructions

Format	Number of APC Instruction Fields	Definition	Description
" ="	1	Direct store	Operand placed in the destination location
" LS n"	2	End-off left shift and store	Operand left-shifted n bits (n ≤ CMW) and placed in the destination location
" RS n"	2	End-off right shift and store	Operand right-shifted n bits (n ≤ CMW) and placed in the destination location
"JMPZ"	1	Jump if zero	If the operand is zero, transfer control to the destination instruction

is a short instruction with label "A " which takes the contents of CMEM location 18 and gates them onto the COMP. As another example

```
column
  1
  ↓
 "B    CMD   4  LS   15 CMI    3",
```

is a short instruction with label "B " which takes the contents of CMEM location 4, left shifts them 15 bits, and gates the result to the CMEM word whose address is in CMEM location 3. Finally,

```
column
  1
  ↓
" C   COMPJMPZA    ",
```

is a short instruction which results in instruction "A " being executed next if the contents of the COMP are zero. If not, the instruction immediately following instruction " C " is executed next. (Sequential execution of instructions is standard.)

3.2.4 Long Instructions

A long APC instruction involves two operands, two operations, and one destination. It may be thought of as a short instruction which has the single operand replaced by the result of an operation involving two operands. References to operands and destinations are made as for short instructions. The format of a long instruction is an operand (OPR1) followed by a long operation (LOP) followed by another operand (OPR2) followed by a short operation and then a destination, symbolically OPR1.LOP.OPR2.SOP.DEST. Thus, the operand of a short instruction is replaced by OPR1.LOP.OPR2. The definitions and formats of the long operations are given in Table 6.

Thus

```
column                                        column
  1                                             43
  ↓                                             ↓
" STR   IB AND IMD      =COMP    ,
                                               523"
```

is a long instruction with label " STR" which logically AND's the contents of the IB and the immediate constant 523_8 or 101010011_2 and gates the result onto the COMP. This instruction also results in replacement of the IB contents with those of a card read by the host computer system, as described in Section 2. To take another example,

```
column
  1
  ↓
"       CMD    5    – CMD    4JMPZ STR"
```

is an unlabeled long instruction which subtracts the contents of CMEM location 4 from location 5 and transfers to instruction " STR" if the result is zero.

Table 6
Long APC Instructions

Format	Name	Result
" AND"	Logical AND	Bitwise logical AND of OPR1 and OPR2
" OR"	Logical OR	Bitwise logical OR of OPR1 and OPR2
" NOT"	Logical COMPLEMENT	Bitwise logical COMPLEMENT of OPR2 (OPR1 is ignored but must be present)
" EOR"	Logical EXCLUSIVE OR	Bitwise logical EXCLUSIVE OR of OPR1 and OPR2
" +"	Add	Arithmetic sum of OPR1 and OPR2 (OPR2 is masked by CMSK; carries are inhibited by CSDR, described in Section 2)
" –"	Subtract	OPR1 minus OPR2 (arithmetic difference) (OPR2 masked by CMSK; carries are inhibited by CSDR, described in Section 2)

3.2.5 Unconditional Transfer and Termination

The "GOTO" instruction is an unconditional APC transfer. "GOTO" is placed in the first APC instruction field (columns 5 through 8) and is followed by the instruction label of the transfer destination. For example

```
column
  1
  ↓
  "BAK GOTO STR"
```

is an APC instruction with label "BAK " which results in an unconditional transfer to instruction " STR".

The "STOP" instruction (columns 5 through 8) results in termination of the simulation following execution of the AP portion of this instruction (if any).

3.3 AP Instructions

The AP portion of an instruction is placed in columns 45 through 71. It is organized in five fields of one column each followed by 11 fields of two columns each. Each AP instruction fills the first five columns and as many of the two column fields (working left to right) as necessary. There should be no blank fields within the instruction. The field following the final nonblank AP field should contain a right-justified comma (" ,") if the entire AP portion has not been used.

3.3.1 Nonstandard Horizontal Inputs

Columns 45 through 49 contain the nonstandard horizontal inputs A_1, A_2, B_1, B_2, and B_3 respectively. The A's specify the nonstandard $\hat{X}$ control according to the code in Table 7. The B's specify the nonstandard y' control according to the code in Table 8.

Table 7
Code for the Nonstandard $\hat{X}$ Control

Input		Description
A_1	A_2	
0	0	$\hat{X} = X$ (bit-slice search)
0	1	$\hat{X} = X$
1	0	$\hat{X} = Y'$
1	1	$\hat{X} = Y$

Table 8
Code for the Nonstandard Y′ Control

Input			Description
B_1	B_2	B_3	
0	0	0	$Y_i' = X_i$
0	0	1	$Y_i' = \overline{Y}_i$
0	1	0	$Y_i' = Y_{i-1}$ (left shift)
0	1	1	$Y_i' = Y_{i+1}$ (right shift)
1	0	0	$Y_i' = X_i \cdot Y_i$ (and)
1	0	1	$Y_i' = X_i + Y_i$ (or)
1	1	0	$Y_i' = X_i \oplus Y_i \oplus$ carry (sum)
1	1	1	$Y_i = X_i \oplus Y_i$

3.3.2 Other Control Line Inputs

The remaining portion of the AP instruction consists of up to eleven mnemonic specifications for setting the various RS, RRN, and AMOB control lines. The different codes and their meanings are given in Table 9. The codes may appear in any order. If no MODE line is specified, MODE 0 is assumed. If both RD and GB are specified, the result of the logical OR of these instructions is gated onto the AMOB. The use of VO eliminates the need for most of the other specifications, which are related to the normal RS control structure.

As an example the AP instruction

Table 9
AP Instruction Mnemonics*

Mnemonic Code†	Meaning	Description
{MO	Mode zero	Sets the MODE line to "0". The standard horizontal inputs are $\hat{X} = X$, $Y' = X$.
M1}	Mode one	Sets the MODE line to "1". The standard horizontal inputs are $\hat{X} = X$, $Y' =$ sum.
CN	Conventional	Sets the CONV line to "1" (otherwise CONV = 0).
ML	Multi	Sets the MULTI line to "1" (otherwise MULTI = 0).
{FC	Full word control	Sets the RSFULL line to "1". The standard horizontal inputs are replaced by nonstandard inputs for words responding to the full word search.
RC}	Response store control	Sets the RSFULL line to "0". The standard horizontal inputs are replaced by nonstandard inputs for words responding to the RS search.
SR	Search	Sets the SEARCH line to "1". The RB is set for words responding to the full word search (otherwise SEARCH = 0)
RD	Read	Sets the READ line to "1"; reads the AP word whose address is in the AMAR into the AMOB (otherwise READ = 0).
GB	Gate bottom	The $\hat{X}$ output of the bottom AP word is gated to the AMOB on the next clock.
GR	Gate response	The RRN gates onto the AMAR on the next clock the biggest address of any word with the response output bit (ROB) set.
RB	Reset response bit	The ROB of the AP word with its address in the AMAR is set to "0" on the next clock.
CR	Clock on response	Sets the CLKRESP line to "1". The clock is applied to conventionally addressed words (CN) and to words whose standard horizontal inputs have been changed (ML).
CB	Reset clock bit	All clock bits (least significant RS bit) are set to "0" on the next clock.
VO	Vertical override	Overrides the normal RS control structure; the HCR contents are used for the horizontal control inputs to each AP word.

*Reference 2 gives full details on these instructions.
†Brackets indicate that only one of the enclosed codes should be used.

```
                    column
                      45
                      ↓
                    "00110MLFCCRSRRD ,"
```

results in the COMP being added to those AP words responding to the full word search criteria given by the combination of the COMP, SC1, and SC2 ("MLFCCR"). Futhermore ("SR") the response bit (RB) specified by the RS sections of the COMP and SC1 is set for those words satisfying the search criteria. Finally ("RD") the AP word whose address is in the AMAR is gated onto the AMOB.

3.4 Utility Instructions

A number of non-APC-executable utility instructions have been added to the mnemonic APC/AP language for the convenience of the simulation user. Utility instructions are placed within the APC portion of an instruction card. An instruction label and an AP portion may be included, but a utility instruction may not be combined with any of the APC instructions defined in Section 3.2. Utility instructions are divided into two groups: memory dumps and statistics dumps.

3.4.1 Memory Dumps

There are five memory dump instructions:

DMCM	– dumps the contents of CMEM,
DMSR	– dumps the contents of SREG,
DMAP	– Dumps the contents of the RS and the APM (the RS and APM contents are separated and spaces are inserted between AP syllables, according to the current SDR, if there is sufficient room),
DMHR	– dumps the contents of HCR,
DMAL	– dumps the contents of all data memories (CMEM, SREG, AP, and HCR).

The dump instructions should be placed in the first APC instruction field (columns 5 through 8), followed by a right-justified comma in the second APC field ("   ,") unless the instruction card has an AP portion.

3.4.2 Statistics Dumps

Statistics dump instructions require two APC instruction fields in the format

```
                    column
                      5
                      ↓
                    "STAT    m",
```

where m = 1, 2, 3, 4. The instruction may have a label and an AP portion. If there is no AP portion, the statistics instruction should be followed by a right-justified comma in the third APC field (columns 13 through 16). A "STAT" instruction results in the printing of a set of APC/AP program statistics selected by the value of m. The statistics printed are those accumulated within the program segment beginning with the previous "STAT" instruction (or the beginning of the program, if none) and ending with present "STAT" instruction. To eliminate confusion in the interpretation of "STAT" outputs, *all* output lists are reset to zero following the execution of *any* of the four possible "STAT" instructions. Cumulative statistics are updated each time the output lists are reset. The full set of cumulative statistics is automatically printed following the "STOP" instruction. All utility instructions are ignored in the accumulation of APC/AP program statistics.

The instruction "STAT 2" results in the printing of the following basic program statistics for the segment in question:

- Number of instructions and number executed,
- Number of instructions with immediate constants and the number executed,
- Number of APC instructions executed,
- Number of short instructions executed and how many of these did not have an AP portion,
- Number of long instructions executed and how many of these did not have an AP portion,
- Number of "GOTO" instructions executed and how many of these did not have an AP portion,
- Number of executed instructions without an AP portion,
- Number of executed instructions with an AP portion,
- Number of executed instructions with both APC and AP portions.

The instruction "STAT 3" results in the printing of the following detailed APC statistics for the segment in question:

- Number of "GOTO" transfers made and the number that transferred to within three instructions away,
- Number of "JMPZ" transfers made and the number that transferred to within three instructions away,
- Number of times the input buffer was accessed,
- Number of times the output buffer was used as a destination register,
- Number of references to special registers and the number of these references that were indirect,
- Number of references to the controller memory and the number of these references that were indirect,
- Average shift length,
- Number of times each long and short APC operation was executed.

The instruction "STAT 4" results in the printing of the following detailed AP statistics for the segment in question:

• Number of AP instructions executed

• Average and maximum number of response resolution bits (RRB) that were set following a search instruction ("SR").

• Number of times each set of nonstandard horizontal inputs was specified in an AP instruction (printed separately for MODE 0, MODE 1, and the combination of the two)

• Number of times each different combination of AP mnemonics was used in an instruction (printed separately for MODE 0, MODE 1, and the combination of the two)

The instruction "STAT 1" results in the printing of all of the above statistics (m = 2, 3, 4) for the segment in question.

To clarify the use of "STAT" instructions, consider the program

```
                    column
                      1
                      ↓
              {  "BEGN        IB   = CMD   0   ,
  segment 1   {                •
              {                •
                               •
                             STAT   2      ,
              {                •
  segment 2   {                •
              {                •
                             STAT   4      ,
              {                •
  segment 3   {                •
              {                •
                             STAT   3      ,
                               •
                               •
                               •
                             STOP    ,"
```

This program will output basic program statistics for segment 1, detailed AP statistics for segment 2, and detailed APC statistics for segment 3, and complete statistics (m = 2, 3, 4) for the entire program.

4. Deck Organization

The data for the simulation program is organized in three sections:

1. Simulation size parameters (one card),
2. APC/AP program (mnemonic codes),
3. APC data (to be read into the IB during the APC program execution).

Each of the simulation size parameters (Table 3) should be placed on the first card as a right-justified integer in the following order:

columns	parameter
1-5	APL
6-10	APMW
11-15	RSW
16-20	BYTSIZ
21-25	CMW
26-30	CML

The complete deck organization (including the simulation program prepared for running on the CDC 3800) is shown in Fig. 2.

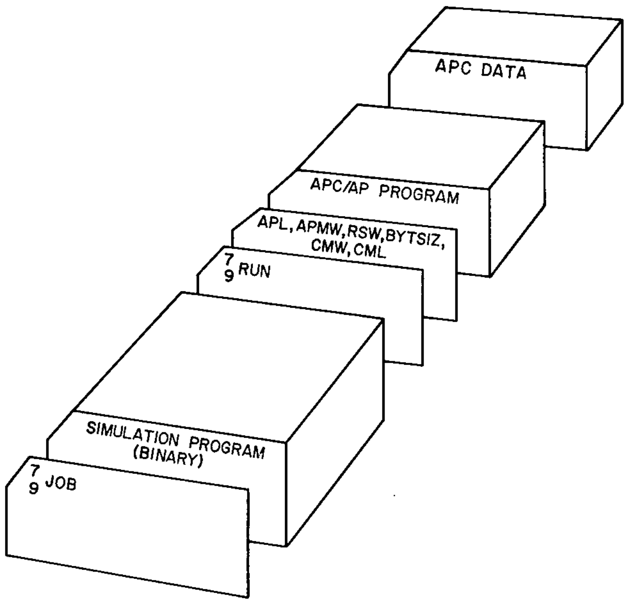


Fig. 2 — Deck configuration

5. Program Output

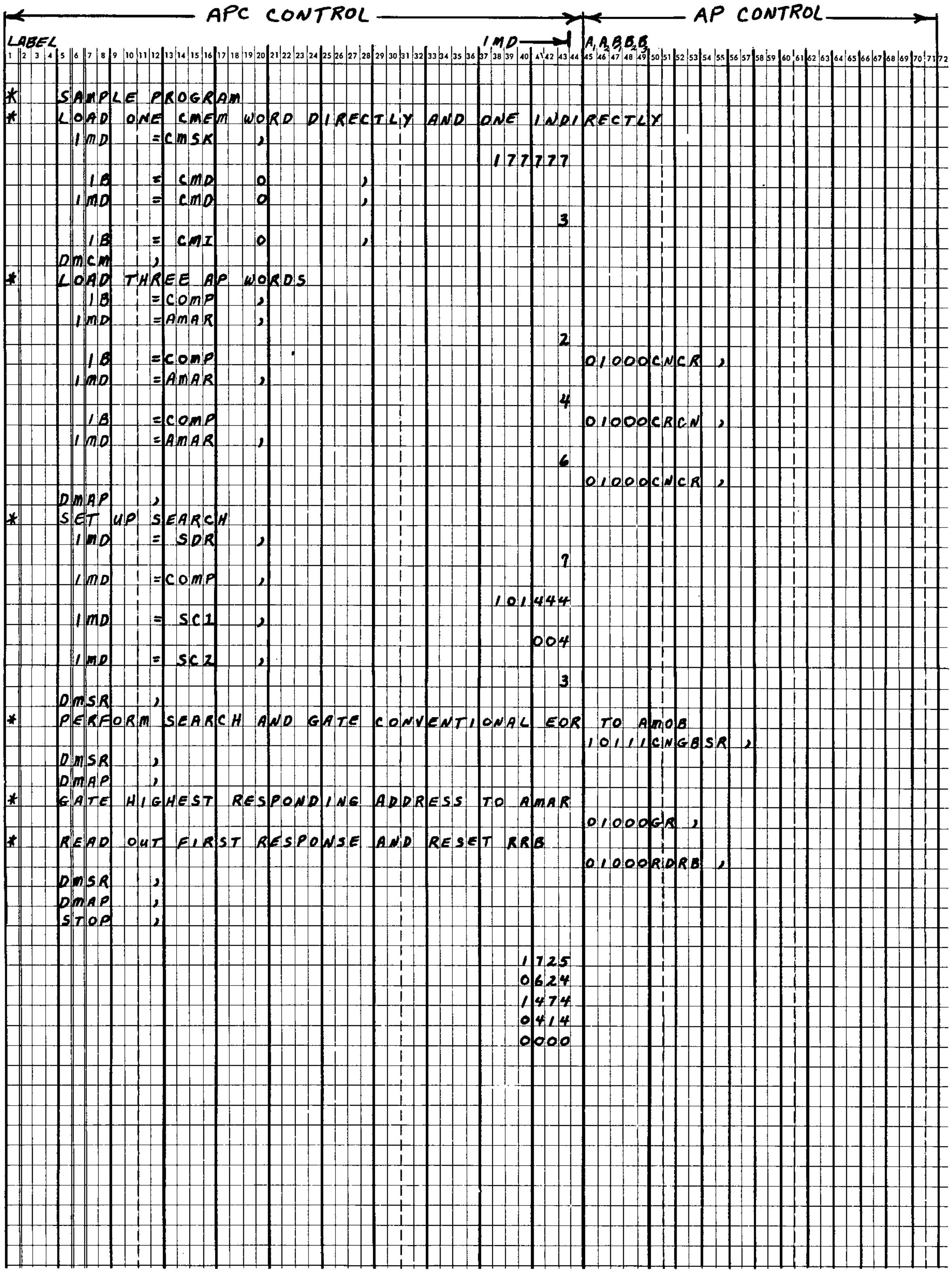
The simulation program produces the following printout:

1. A listing of the APC/AP program with instructions numbered and mnemonics separated for easy reading,
2. Outputs from the APC OB (printed as binary numbers),
3. APC inputs obtained through the IB (printed as binary numbers),
4. Utility dumps.

During execution the program checks for common APC/AP programming errors (such as incorrect format and numbers out of bounds). If an error is detected, a message giving its exact location is printed and execution is terminated.

6. Sample Program

To aid in the use of the simulation program, a complete example will be given. The sample program inputs data through the IB, loads CMEM and AP locations, and performs a number of AP functions. The coding sheet used in the program preparation is shown in Fig. 3. The



```
←———————— APC CONTROL ————————————————————→|←———— AP CONTROL ————————→
LABEL                               IMD ——→|A1A2B1B2B3
*   SAMPLE PROGRAM
*   LOAD ONE CMEM WORD DIRECTLY AND ONE INDIRECTLY
     IMD   =CMSK   ,
                                     177777
      IB   = CMD   0       ,
     IMD   = CMD   0       ,
                                          3
      IB   = CMI   0       ,
    DMCM        ,
*   LOAD THREE AP WORDS
      IB   =COMP   ,
     IMD   =AMAR   ,
                                          2
      IB   =COMP                            01000CNCR ,
     IMD   =AMAR   ,
                                          4
      IB   =COMP                            01000CRCN ,
     IMD   =AMAR   ,
                                          6
                                            01000CNCR ,
    DMAP        ,
*   SET UP SEARCH
     IMD   = SDR   ,
                                          7
     IMD   =COMP   ,
                                     101444
     IMD   = SC1   ,
                                        004
     IMD   = SC2   ,
                                          3
    DMSR        ,
*   PERFORM SEARCH AND GATE CONVENTIONAL EOR TO AMOB
                                            10111CNGBSR ,
    DMSR        ,
    DMAP        ,
*   GATE HIGHEST RESPONDING ADDRESS TO AMAR
                                            01000GR ,
*   READ OUT FIRST RESPONSE AND RESET RRB
                                            01000RDRB ,
    DMSR        ,
    DMAP        ,
    STOP        ,

                                       1725
                                       0624
                                       1474
                                       0414
                                       0000
```

Fig. 3 – Coding sheet for a sample program

input deck is listed in Fig. 4, and the complete printout resulting from execution of the simulation program is included in Appendix A. The following simulation size parameters were chosen: APL = 8, APMW = 12, RSW = 4, BYTSIZ = 4, CMW = 16, and CML = 8.

The original IB contents are zero, and the first reference to the IB (instruction 2) serves only to load the IB with the first input card (1725_8). Also, although only four input cards are of interest, a final dummy must be included, since it will be read when the last input data of interest (414_8) is gated from the IB (instruction 10).

The AP search instruction (instruction 19) specifies exact match for the first syllable, greater than or equal to for the second syllable, and less than or equal to for the third (syllables counted right to left). The same instruction gates the EXCLUSIVE OR of the COMP and

```
     8   12     4    4   16     8
*    SAMPLE PROGRAM
*    LOAD ONE CMEM WORD DIRECTLY AND ONE INDIRECTLY
      IMD   =CMSK   ,
                                      177777
       IB   = CMD   0   ,
      IMD   = CMD   0   ,
                                           3
       IB   = CMI   0   ,
     DMCM   ,
*    LOAD THREE AP WORDS
       IB   =COMP   ,
      IMD   =AMAR   ,
                                           2
       IB   =COMP                            01000CNCR ,
      IMD   =AMAR   ,
                                           4
       IB   =COMP                            01000CRCN ,
      IMD   =AMAR   ,
                                           6
                                             01000CNCR ,
     DMAP   ,
*    SET UP SEARCH
      IMD   = SDR   ,
                                           7
      IMD   =COMP   ,
                                      101444
      IMD   = SC1   ,
                                         004
      IMD   = SC2   ,
                                           3
     DMSR   ,
*     PERFORM SEARCH AND GATE CONVENTIONAL EOR TO AMOB
                                             10111CNGBSR ,
     DMSR   ,
     DMAP   ,
*    GATE HIGHEST RESPONDING ADDRESS TO AMAR
                                             01000GR ,
*    READ OUT FIRST RESPONSE AND RESET RRB
                                             01000RDRB ,
     DMSR   ,
     DMAP   ,
     STOP   ,
                                        1725
                                        0624
                                        1474
                                        0414
                                        0000
```

Fig. 4 – Listing of the program coded as shown in Fig. 3

AP location 6 onto the AMOB. The next AP instruction (instruction 22) gates the highest address of the responding words (location 4) onto the AMAR and is followed by an AP instruction that resets the RRB of this location while reading its contents out to the AMOB. Program execution may be traced by viewing the memory dumps. The program required 62 seconds to execute.

REFERENCES

1. Shore, J.E., and Polkinghorn, F.A., "A Fast, Flexible, Highly Parallel Associative Processor," NRL Report 6961, Nov. 28, 1969.
2. Shore, J.E., and Collins, T.L., "Another Associative Processor," NRL Report 7348, November 1971.

APPENDIX A
PRINTOUT FROM THE SAMPLE PROGRAM

```
       MICROPROGRAM LISTING



    *    SAMPLE PROGRAM
    *    LOAD ONE CMEM WORD DIRECTLY AND ONE INDIRECTLY
  1        IMD     = CMSK     ,
                                        177777
  2         IB     =  CMD    0    ,
  3        IMD     =  CMD    0    ,
                                             3
  4         IB     =  CMI    0    ,
  5       DMCM     ,
    *    LOAD THREE AP WORDS
  6         IB     = COMP     ,
  7        IMD     = AMAR     ,
                                             2
  8         IB     = COMP                                      01000   CN   CR    ,
  9        IMD     = AMAR     ,
                                             4
 10         IB     = COMP                                      01000   CR   CN    ,
 11        IMD     = AMAR     ,
                                             6
 12                                                            01000   CN   CR    ,
 13       DMAP     ,
    *    SET UP SEARCH
 14        IMD     =  SDR     ,
                                             7
 15        IMD     = COMP     ,
                                        101444
 16        IMD     =  SC1     ,
                                           004
 17        IMD     =  SC2     ,
                                             3
 18       DMSR     ,
    *     PERFORM SEARCH AND GATE CONVENTIONAL EOR TO AMOB
 19                                                            10111   CN   GB   SR    ,
 20       DMSR     ,
 21       DMAP     ,
    *    GATE HIGHEST RESPONDING ADDRESS TO AMAR
 22                                                            01000   GR    ,
    *    READ OUT FIRST RESPONSE AND RESET RRB
 23                                                            01000   RD   RB    ,
 24       DMSR     ,
 25       DMAP     ,
 26       STOP     ,


 BEGIN EXECUTION
```

```
    INPUT =
0000001111010101

    INPUT =
0000000110010100
```

```
                          MPC MEMORY DUMP AT IC    5

   0 0000000000000011
     0000000000000000
     0000000000000000
     0000001111010101
     0000000000000000
   5 0000000000000000
     0000000000000000
     0000000000000000

    INPUT =
0000001100111100

    INPUT =
0000000100001100

    INPUT =
0000000000000000
```

```
                          AP DUMP AT IC =  13

   0 0000   000000000000
     0000   000000000000
     0000   000110010100
     0000   000000000000
     0000   001100111100
   5 0000   000000000000
     0000   000100001100
     0000   000000000000
```

```
                          SPECIAL REGISTER DUMP AT IC =  18


             111   SDR
         0000100   SCl
             011   SC2
             000    CM
             110  AMAR
            0000  NRSP
1000001100100100  COMP
    000000000000  AMOB
0000000000000000  CSDR
1111111111111111  CMSK
0000000000000000    IB
0000000000000000    OB
```

```
                          SPECIAL REGISTER DUMP AT IC =  20


             111   SDR
         0000100   SCl
             011   SC2
             000    CM
             110  AMAR
            0000  NRSP
1000001100100100  COMP
    001000101000  AMOB
0000000000000000  CSDR
1111111111111111  CMSK
0000000000000000    IB
0000000000000000    OB
```

```
                       AP DUMP AT IC =  21


0 0000   0000 0000 0000
  0000   0000 0000 0000
  1000   0001 1001 0100
  0000   0000 0000 0000
  1000   0011 0011 1100
5 0000   0000 0000 0000
  0000   0001 0000 1100
  0000   0000 0000 0000
```

```
                         SPECIAL REGISTER DUMP AT IC =  24

                 111  SDR
             0000100  SC1
                 011  SC2
                 000   CM
                 100 AMAR
                0010 NRSP
    1000001100100100 COMP
        001100111100 AMOB
    0000000000000000 CSDR
    1111111111111111 CMSK
    0000000000000000   IB
    0000000000000000   OB
```

```
                      AP DUMP AT IC =  25

0 0000  0000 0000 0000
  0000  0000 0000 0000
  1000  0001 1001 0100
  0000  0000 0000 0000
  0000  0011 0011 1100
5 0000  0000 0000 0000
  0000  0001 0000 1100
  0000  0000 0000 0000
```

```
                    BASIC STATISTICS FOR ENTIRE PROGRAM
                   ************************************


NUMBER OF MICROINSTRUCTIONS IN SEGMENT =    18      NUMBER EXECUTED =     18

NUMBER OF MICROINSTRUCTIONS IN SEGMENT WITH INTERNAL CONSTANTS =     9      NUMBER EXECUTED =      9

TOTAL NUMBER OF MPC INSTRUCTIONS EXECUTED =     14

     TOTAL NUMBER OF SHORT INSTRUCTIONS EXECUTED =     14      NUMBER NOT INVOLVING THE AP=     12

     TOTAL NUMBER OF LONG INSTRUCTIONS EXECUTED =      0      NUMBER NOT INVOLVING THE AP=      0

     TOTAL NUMBER OF -GO TO- TRANSFERS EXECUTED =      0          NUMBER NOT INVOLVING THE AP =      0

TOTAL NUMBER OF EXECUTED INSTRUCTIONS NOT INVOLVING THE AP =     12
TOTAL NUMBER OF AP INSTRUCTIONS EXECUTED =     6
TOTAL NUMBER OF COMBINED (AP/MPC) INSTRUCTIONS EXECUTED =     2
```

```
                    DETAILED MPC STATISTICS FOR ENTIRE PROGRAM
                   *************************************************

NUMBER OF -GO TO- TRANSFERS MADE =       0   NUMBER WITHIN THREE INSTRUCTIONS AWAY =       0

NUMBER OF -JMPZ- TRANSFERS MADE =       0   NUMBER WITHIN THREE INSTRUCTIONS AWAY =       0

NUMBER OF TIMES THE INPUT BUFFER WAS ACCESSED =       5    NUMBER OF TIMES DATA GATED TO OUTPUT BUFFER =       0

NUMBER OF WORKING REGISTER REFERENCES =      16     NUMBER INDIRECT =       0

NUMBER OF CONTROL MEMORY REFERENCES =       3     NUMBER INDIRECT =       1

THE AVERAGE SHIFT LENGTH WAS   0.0 BITS


               THE MPC OPERATIONS DURING THIS SEGMENT WERE AS FOLLOWS

    JMPZ         =         LS         RS         AND        OR        NOT        EOR        +          -
      0         14          0          0          0          0          0          0          0          0
```

```
                    COMPLETE AP STATISTICS FOR ENTIRE PROGRAM
                    *********************************************


   TOTAL NUMBER OF AP INSTRUCTIONS EXECUTED =     6

   AVERAGE NUMBER OF RESPONSE RESOLUTION BITS SET FOLLOWING SEARCH =   2.0      MAXIMUM =   2

   NUMBER OF TIMES THE CLOCK BIT SLICE WAS USED TO CLOCK THE AP =     0

MODE ZERO STATISTICS
********************

                      NON-STANDARD SPECIFICATIONS


     YPRIME =          X        .NOT.Y       BR         BL       X.AND.Y   X.OR.Y      SUM      X.EOR.Y
*************************************************************************************************************
XHAT = X (BSS)  *      0          0          0          0          0          0          0          0
XHAT = X        *      5          0          0          0          0          0          0          0
XHAT = YPRIME   *      0          0          0          0          0          0          0          1
XHAT = Y        *      0          0          0          0          0          0          0          0


               CONTROL INPUTS


NUMBER  TYPE

  3     CLOKONRS  CONVENT
  1      SEARCH  GATEBOTM  CONVENT
  1     GATERESP
  1     RESETRRB   READ
```

```
COMBINED MODE STATISTICS
************************


                            NON-STANDARD SPECIFICATIONS


       YPRIME =          X        .NOT.Y      BR         BL       X.AND.Y    X.OR.Y      SUM      X.EOR.Y
    *******************************************************************************************************
    XHAT = X (BSS) *     0          0          0          0          0          0          0          0
    XHAT = X       *     5          0          0          0          0          0          0          0
    XHAT = YPRIME  *     0          0          0          0          0          0          0          1
    XHAT = Y       *     0          0          0          0          0          0          0          0


                   CONTROL INPUTS


NUMBER  TYPE

  3     CLOKONRS  CONVENT
  1      SEARCH  GATEBOTM  CONVENT
  1     GATERESP
  1     RESETRRB   READ
```

```
MODE ONE STATISTICS
*******************


                    NON-STANDARD SPECIFICATIONS


    YPRIME =         X       .NOT.Y       BR         BL       X.AND.Y     X.OR.Y       SUM      X.EOR.Y
**********************************************************************************************************
XHAT = X (BSS) *      0          0          0          0          0           0          0          0
XHAT = X       *      0          0          0          0          0           0          0          0
XHAT = YPRIME  *      0          0          0          0          0           0          0          0
XHAT = Y       *      0          0          0          0          0           0          0          0


                CONTROL INPUTS


NUMBER  TYPE
```

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Naval Research Laboratory Washington, D.C. 20390	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

SOFTWARE SIMULATION OF AN ASSOCIATIVE PROCESSOR

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*
An interim report on a continuing NRL Problem

5. AUTHOR(S) *(First name, middle initial, last name)*

John E. Shore

6. REPORT DATE December 16, 1971	7a. TOTAL NO. OF PAGES 32	7b. NO. OF REFS 2
8a. CONTRACT OR GRANT NO. NRL Problem R06-41 b. PROJECT NO. WF08-151-702 and WF15-241-601/602 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) NRL Report 7351 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*	

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Department of the Navy Naval Air Systems Command Washington, D.C. 20360

13. ABSTRACT

A software simulation of an associative processor (AP) has been developed in Fortran as a tool for use in some research projects at NRL. The particular AP design being simulated is that described in NRL Report 7348, a design oriented towards the requirements of the Advanced Avionic Digital Computer (AADC) under development by the Naval Air Systems Command.

The simulated AP is driven by a simulated associative processor controller (APC). The APC is a simple, sequential computer with an instruction memory, a data memory, an arithmetic and control section, and a set of special, AP associated registers. The entire AP/APC simulation is at the bit level with the exception of the APC instruction memory, which contains a mnemonic APC code written by the user.

In this report the overall simulation is described and instructions for its use are given. A complete example is included.

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Computers						
Design						
Associative processors						
Simulation						
Software						
Programming						
Parallel processing						
Array processing						

DD FORM 1 NOV 65 1473 (BACK)
(PAGE 2)

Security Classification